AD-A007 922

BATCH COMPUTER SCHEDULING: A HEURISTICALLY MOTIVATED APPROACH

Stephen R. Kimbleton

University of Southern California

Prepared for:

Office of Naval Research
Advanced Research Projects Agency

September 1974

DISTRIBUTED BY:

NTIS
National Technical Information Service
U. S. DEPARTMENT OF COMMERCE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) BATCH COMPUTER SCHEDULING: A HEURISTICALLY MOTIVATED APPROACH		5. TYPE OF REPORT & PERIOD COVERED Research Paper
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Stephen R. Kimbleton		8. CONTRACT OR GRANT NUMBER(s) N00014-67-A-0181-0036 and DAHC 15 72 C 0308
9. PERFORMING ORGANIZATION NAME AND ADDRESS University of Southern California Information Sciences Institute 4676 Admiralty Way, Marina del Rey, California 90291		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NR 049-311 ARPA Order #2223/1
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research Information Systems Program Arlington, Virginia (over)		12. REPORT DATE September 1974
		13. NUMBER OF PAGES 13
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Reproduction in whole or in part is permitted for any purpose of the United States Government

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

D D C
RECEIVED
APR 15 1975
B

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Reproduced by
NATIONAL TECHNICAL
INFORMATION SERVICE
US Department of Commerce
Springfield, VA. 22151

18. SUPPLEMENTARY NOTES

Presented at the Second Annual SIGMETRICS Symposium on Measurement and Evaluation, Montreal, Canada, September/October 1974.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Job scheduling, performance prediction, simulation, analytic models, system design, performance analysis

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Efficient scheduling of jobs for computer systems is a problem of continuing concern. The applicability of scheduling methodologies described in the operation research literature is severly restricted by the dimensionality of job characteristics, the number of distinct resource types comprising a computer system, the non-deterministic nature of the system due to both interprocess interaction and contention, and the existence of a multitude of constraints affecting job initiation times, job completion times, and job interactions. In view of the large number of issues which must be considered in job scheduling, a heuristic approach seems (cont)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

PRICES SUBJECT TO CHANGE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD A007922

111129

11. Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington, Virginia 22209

20 (cont.)
appropriate. This paper describes an initial implementation of such an approach based upon a fast, analytically driven, performance prediction tool.

BATCH COMPUTER SCHEDULING: A HEURISTICALLY MOTIVATED APPROACH*

By

Stephen R. Kimbleton
USC/Information Sciences Institute
4676 Admiralty Way
Marina del Rey, California 90291

ABSTRACT

Efficient scheduling of jobs for computer systems is a problem of continuing concern. The applicability of scheduling methodology described in the operations research literature is severely restricted by the dimensionality of job characteristics, the number of distinct resource types comprising a computer system, the non-deterministic nature of the system due to both interprocess interaction and contention, and the existence of a multitude of constraints effecting job initiation times, job completion times, and job interactions. In view of the large number of issues which must be considered in job scheduling, a heuristic approach seems appropriate. This paper describes an initial implementation of such an approach based upon a fast, analytically driven, performance prediction tool.

1. INTRODUCTION

Job scheduling is an issue of continuing concern to computer center management. This concern reflects, in large part, the inapplicability of the substantial body of work on scheduling existing within the operations research literature [BAKEK 71], [BELLL 70], [CONWR 67], [SAHNV 72]. This inapplicability is due to the fact that most of the operations research literature on scheduling is concerned with either deterministic scheduling, which is based on mathematical programming techniques, or probabilistic scheduling, which is based on queuing theory. However, a computer system contains both deterministic and probabilistic components. Further, some devices, e.g., tapes and disks, may alternate between shared and serially

----

*Preparation of this paper was supported by the Advanced Research Projects Agency under Contract No. DAHC15 72 C 0308, ARPA Order No. 2223, Program Code No. 3D30 and 3P10, and the Office of Naval Research, Information Systems Program under Contract N00014-67-A-0181-0036 (NR 049-311).

The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Advanced Research Projects Agency, the Office of Naval Research, or the U.S. Government.

reusable status. Hence the device characteristics can have deterministic status changes and probabilistic service times. It follows that neither deterministic nor probabilistic approaches can be used in isolation, and attempts to do so usually result in information which provides a great deal of insight but little practical guidance in constructing actual schedules for the execution of jobs on computer systems.

The evident inability of analytic approaches to handle the complexity of computer system scheduling suggests the desirability of developing heuristic techniques for job scheduling. Such techniques have been used in a wide variety of contexts within computer engineering; a notable success is their utilization in developing 'good' topologies for ARPANET [FRANH 70, 72].

Development of a heuristic approach to computer systems scheduling requires:

1. Prediction of the performance of a given computer system processing a specified workload in accord with a given schedule,

2. Identification of a means for comparing two schedules and improving a given schedule, and

3. Determination of when the iteration cycle implicitly defined by (1) and (2) should stop.

A fundamental difficulty in developing computer system schedules has been the cost implicit in (1). That is, the relative desirability of the schedule is related to device utilizations and delays. However, simulators which are sufficiently detailed to provide useful information on device delays and utilizations usually exhibit execution times which tend to be in the lower end of the range 1-10 times faster than real time. Thus, heuristic schedule generation based on simulators is rendered infeasible.

In [KIMBS 74A], a very fast analytically driven approach to computer system performance prediction for production batch jobs has been described. This tool seems appropriate to (1) and, as will be discussed later, the class of jobs for which it is applicable appears to be the class of jobs for which the development of schedules is most appropriate. Thus, the basic requirement for developing a heuristic approach to scheduling is at hand. In the remainder of this paper, we shall identify a means forcomparing schedules and discuss an initial heuristic approach. An example of the output from this approach is given in Section 4.

This paper is not intended to be a contribution to either scheduling theory, per se, or to computer system modeling. Rather, it is intended to demonstrate that the proper combination of theoretical knowledge from each of these two areas can provide a practical means of solving a problem of continuing concern to computer center managers in a cost effective manner. Further, without such a theoretically based, practically oriented approach, a usable solution to this problem appears remote.

2. COMPUTER SYSTEM SCHEDULE COMPONENTS

In this section we define what is meant by computer system schedule, identify a class of jobs most appropriate to scheduling, describe constraints to be observed in constructing a schedule, and identify a quantitative means for schedule comparison.

Job Types

The jobs to be processed by a computer system can be divided into four categories, depending upon whether or not times of arrival and resource requirements are known. External scheduling is most appropriate for jobs whose arrival times and resource requirements are known. In the remainder of this paper, we are concerned with this class of jobs termed the production batch jobs.

Schedule Definition

Label the collection of jobs to be processed during a shift 1,...,N. A schedule is a permutation P(1),...,P(N), of these indices describing the sequence in which jobs are to be initiated. Jobs will be initiated in the (left to right) order indicated subject to: (1) job P(1) is initiated first, (2) once a job is initiated, that job remains resident in the system until it is completed, i.e. its execution is not involuntarily suspended, (3) if jobs P(1),...,P(j) have been initiated, the next job to be initiated will be job P(j+1), and (4) if jobs P(1),...,P(j-1) have been initiated, job P(j) will be initiated immediately once all constraints affecting its initiation have been satisfied (cf. the subsection below discussing constraints).

Schedule Constraints

Any permutation P(1),...,P(N) constitutes a schedule. A schedule which satisfies all constraints is termed feasible. For batch production oriented schedules, four major types of constraints exist: priorities, deadlines, precedence relations and resource requirements. For simplicity we shall assume that all jobs have the same priority and consist of one jobstep. The required extensions to incorporate multiple (operational) priority classes and jobs consisting of multiple jobsteps will be evident once the general approach has been described. We also assume that if a precedence relation exists between jobs P(j) and P(k), $j < k$, then job P(j) must be completed before job P(k) can be initiated. Finally, we shall assume that setup/teardown times are negligible.

Comparing Computer System Schedules

For a production batch computer system, the natural objective function is to minimize usage of system resources subject to the constraint that all deadlines are met. (Note that this objective function also permits incorporation of turnaround time limitations through development of artificial deadlines corresponding to the time of arrival plus acceptable mean turnaround time.) To formulate a quantitative analogue of this concept, we introduce the concept of the shift system reward function (SSRF). This requires introduction of two prior definitions: event and time segment.

DEFINITION. An event is a point in time at which either a job initiation or job termination occurs. A time segment is the time between two successive events.

A schedule containing N jobs will have precisely 2N events and 2N - 1 time segments. The composition of the mix is constant over a time segment and thus it is appropriate to define the time segment system reward function (TSSRF) for a given time segment as:

$$TSSRF = \frac{U(P)C(P) + U(M)C(M) + U(S)C(S) + U(D)C(D)}{C(P) + C(M) + C(D) + C(S) + C(U)}$$

where P,M,D,S,U denote processor, memory, secondary storage, data paths and unit record equipment respectively, U(*) denotes the utilization of the device over the time segment and C(*) denotes the monthly rental cost of the device.

Our objective function can now be formally stated as:

MIN SSRF ($= \sum$ TSSRF)

subject to:

NO MISSED DEADLINES,

where the sum extends over all the time segments comprising the shift.

Note that this objective function attempts to minimize the total cost weighted utilization of the system by the collection of production batch jobs which must be processed during the shift. Thus it tends to ensure that in executing these jobs, the system will be made maximally available (in a cost sense) to the jobs in the other categories, subject to the requirement that all deadlines be met.

In common with almost all optimization techniques, this approach does not assure maximal availability to other job classes since it is possible that a schedule may be found for which some device is effectively operated at the edge of saturation. However, as will be evident from the approach, this can be avoided by a simple modification of the heuristic used to limit the maximum utilization for any given device.

3. SCHEDULE GENERATION

Heuristic generation of a schedule requires characterization of the interaction of a job with a computer system. For production batch jobs, a synthetic module [HAMIP 73], [SREEK 74] characterization proves satisfactory.

Generation of an appropriate schedule now proceeds in two phases: construction of an initial schedule and improvement of a given schedule.

Initial Schedule Generation

Generation of an initial, seizable schedule proves difficult since the schedule determines and is determined by the execution times of the jobs being processed. This impasse is circumvented by initially assigning a (conservative) deterministic processing time to a job. Using this deterministic estimator, a sorting technique can then be used to obtain a feasible schedule, i.e., a schedule needing deadline requirements and satisfying precedence constraints. Treesort [KNUTD 68] appears most appropriate since, in effect, the calculations are analogous to those used in PERT Scheduling. Since the deterministic estimators overestimate job processing times, the initial schedule generator may fail to find a feasible schedule. In this case, one may either terminate the heuristic search, or follow the standard approach of entering the schedule improvement routine.

Schedule Improvement Routine

The development of heuristics for achieving an improved schedule from a given schedule in the context which has been described has received very little study. Accordingly, as a first cut at the problem, a very simple schedule improvement routine has been implemented. This routine in effect constructs the graph of the TSSRF plotted against the time segment number and then attempts to level the 'hills' by filling the 'valleys.' It is widely known that a transposition technique such as described may fail to find an optimal schedule. However, there is reason to think that it is a fairly reasonable technique with a reasonable likelihood of finding a 'good' schedule [LINS 65]. Some results from the application of this technique are given in the following section.

Stopping the Search

Any iterative optimization technique requires the existence of a stopping procedure. At the present state of development of the technique just described, such procedures are clearly inappropriate. Thus, we have chosen to implement a very simple technique which, in effect, decides to stop when either a certain number of iterations have been reached or when the difference between the maximum and minimum TSSRF falls within some specified tolerance limit. Better procedures are needed.

4. AN EXAMPLE

An evaluation of a heuristic simulator should be conducted along two dimensions: detection and agility. Qualitatively, the ability to detect undesirable scheduling characteristics through examination of the time sequence of TSSRFs appears to be satisfactory. Space limitations preclude presentation of a series of examples to support this assertion.

The agility, in a general environment, of the schedule improvement routine remains open to question. It is relatively easy to construct examples in which the agility appears to be either good or bad, depending upon the characteristics of the example. Significant refinement in the heuristics is clearly desirable. The utility of such refinements is evidenced by the ability of an interactive, man-machine version to produce good schedules. In this approach, the initial schedule generator is used to obtain the first schedule, and the 'man' is thereafter responsible for generation of heuristics and determination of when to stop.

The output of the scheduler is divided among five output files representing five different categories of information. Space limitations therefore preclude a complete description of the output for even a simple example. Accordingly, we have chosen to describe the two major input files (job characteristics, system characteristics) to the scheduler for a simple five job example. In this example, the resource requirements of the jobs are identical and the differences are reflected only in the due dates and times of arrival which were chosen to render development of a feasible schedule impossible. Table 3 presents example time segment statistics and Table 4 provides a comparison of the characteristics of the schedule developed by the initial schedule generator and the revised schedule after one pass with the schedule improvement routine. The capability of the scheduler to handle dedicated disks and tapes was not demonstrated due to space limitations. It is hoped that examination of this information will persuade the reader of the potential merits of a heuristic approach. A forthcoming technical report will provide a more substantive basis for this examination.

TABLE 1

SYSTEM CHARACTERISTICS

Number of Drums	2
Number of Pages/Track	6
Drum Rotation Time	35ms.
Number of Disks	8
Number of Pages/Track	3
Disk Rotation Time	25ms.
Minimum Seek Time	10ms.
Maximum Seek Time	60ms.
Average Seek Time	34ms.
Cost Per Unit Time of	
CPU	.2
Core	1.9
Drum	.5
Disk	.03

TABLE 2

JOB CHARACTERISTICS

Time of Arrival	***
Due Date	***
CPU Time Required	40 s.
CPU Time/I/O Operation	40ms.
Primary Memory Required	20pp.
Number of I/O Devices Accessed	10
Drum Access Characteristics	
Probability of Access to Drum I, I=1,2	.333
Pages Transferred Per Access to Drum I, I=1,2	1
Average Latency Per Access to Drum I, I=1,2	17ms.
Probability of Access to Disk I, I=1,...,8	.042
Pages Transferred Per Access to Disk I, I=1,...,8	2
Average Number of Cylinders Per Seek for Disk I, I=1,...,8	100

*** Indicates statistics varied on a per job basis.

TABLE 3

REPRESENTATIVE TIME SEGMENT STATISTICS

Time Segment Beginning	79.75 s.
Time Segment End	178.79 s.
Degree of Multiprogramming	2
Utilizations	
CPU	.51
Core	.40
Drum	.13
Disk	.04
System Reward Function	.195

TABLE 4

COMPARISON OF OUTPUT FROM INITIAL SCHEDULE GENERATOR AND SCHEDULE IMPROVEMENT ROUTINE (ONE PASS)

Schedule Begin	0.0	0.0
Schedule End	328.27	277.83
Total Number Jobs Processed	5	5
Average Degree of Multiprogramming	1.25	1.70
Average I/O Time	36ms.	38ms.
Average Processor Wait Between Two Successive I/O Operations	6ms.	14ms.
SRF	.12	.16
Utilizations		
Core	.52	.50
Primary Memory	.26	.34
Drum	.04	.09
Disk	.01	.03

5. CONCLUDING REMARKS

The preceding discussion is intended to demonstrate that through proper application and combination of theoretical results, an approach to external job scheduling can be achieved which is of practical significance. Clearly, the cost of this approach depends upon the tool used to predict system performance for a given schedule, i.e. the execution speed of ASIM.

In its present version, speeds significantly faster than those otherwise available are achieved (for production batch jobs). Inspection of this version shows that it is highly likely that a revised version with an execution speed which is approximately one quarter of the current version can be achieved.

The heuristic approach to scheduling as described uses existing theoretical scheduling insights in only a limited manner. Clearly, it is desirable to examine the applicability of shortest job, smallest job, and first fit types of heuristics.

6. ACKNOWLEDGMENTS

The author would like to express his appreciation to Professor K. R. Baker whose comments on the general topic of computer system scheduling provided the initial point of departure for this investigation. In addition, significant credit is due to Ms. Claudia Stallings who programmed the initial version of the Initial Schedule Generator and the Schedule Improvement Routine under rather trying circumstances.

REFERENCES

ARTIP 73 Artis, H. Pat, "Interactive Simulation Techniques for Computer System Scheduling," Bell Laboratories, 1973.

BAKEK 71 Baker, K. R., "A Survey of the Literature on the External Scheduling Problem in Data Processing Systems," Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan, 1971.

BELLL 70 Bell Laboratories, "Bibliography: Queuing, Scheduling, and Networks -- Theory and Applications, 1960-1970," Report No. 183, 1971.

CONWR 67 Conway, R. W., W. L. Maxwell and L. W. Miller, Theory of Scheduling, Addison-Wesley, Reading, Massachusetts, 1967.

FRANH 70 Frank, H., I. T. Frisch, and W. Chou, "Topological Considerations in the Design of the ARPA Computer Network," AFIPS Conference Proceedings, 1970 Spring Joint Computer Conference, pp. 581-587.

FRANH 72 Frank, H., R. E. Kahn, and L. Kleinrock, "Computer Communication Network Design--Experience with Theory and Practice," AFIPS Conference Proceedings, 1972 Spring Joint Computer Conference, pp. 255-270.

HAMIP 73 Hamilton, P. A., and B. W. Kernigan, "Synthetically Generated

Performance Test Loads for Operating Systems," Proceedings of the ACM-SIGME Symposium on Measurement and Evaluation, February 26-28, 1973, pp. 121-126.

KIMBS 72 Kimbleton, S. R., "Performance Evaluation--A Structured Approach," AFIPS Conference Proceedings, 1972 Spring Joint Computer Conference, pp. 411-416.

KIMBS 74A Kimbleton, S. R., A Fast Approach to Computer Performance Prediction, USC/Information Sciences Institute, Marina del Rey, California, ISI/RR-74-20, 1974.

KIMBS 74B Kimbleton, S. R., "An Approximate Analytical Technique for Hierarchical Computer System Modeling," USC/Information Sciences Institute, Marina del Rey, California, (in progress).

KNUTD 68 Knuth, D. E., The Art of Computer Programming, Vol. 1, Fundamental Algorithms, Addison-Wesley, 1968.

LINS 65 Lin, S., "Computer Solution of Traveling Salesman Problem," BELL SYSTEM TECHNICAL JOURNAL, Vol. 44, No. 10, December 1965, pp. 2245-69.

SAHNV 72 Sahney, V. K. and J. L. May, Scheduling Computer Operations, Computer and Information Sciences Monograph No. 6, American Institute of Industrial Engineering, Atlanta, Georgia, 1972.

SREEK 74 Sreenivasan, K., and A. J. Kleinman, "On the Construction of a Representative Synthetic Workload," Communications of the ACM, Vol. 27, No. 3, 1964, pp. 127-132.